Computational Clusters Efficient Parallel Data Transmission Paradigm

Mahdi Qasim Mohammed
Mohammed M. Azeez
Mustafa Aljshamee
Distributed high performance computing institute
University of Rostock
Rostock, Germany
mahdi.mohammed@uni-rostock.de
mohammed.azeez1983@yahoo.com
mustafa.aljshamee@uni-rostock.de

Abbas Malekpour
Peter Luksch
Distributed high performance computing institute
University of Rostock
Rostock, Germany
abbas.malekpour@uni-rostock.de
peter.luksch@uni-rostock.de

***Abstract*— Message Passing Interface (MPI) is the most popular and widespread model used for distributed parallel programming in high performance computing systems. Stream Control Transmission Protocol (SCTP) is a standard transport layer protocol. It supports a lot of features not supported by transmission control protocol (TCP), recently the standard transport layer protocol used by MPI, which make SCTP a promising solution to be used as MPI under layer protocol. In this work, Multi-streaming and Multi-homing, the most powerful features supported by SCTP, are used to implement parallel data transmission over computational networks using all the available paths and physical interface cards. For this purpose, an application programming interface API had been designed and used.**

Keywords: Message passing interface (MPI),Stream control transmission protocol (SCTP) .

I. Introduction

The tremendous increase in the utilization of computer networks in different applications and purposes and the raise in performance in networks' features like security, reliability, bandwidth and throughput in the last decades encouraged the computer programmers to benefit these parameters to establish distributed and high performance computing systems. High performance computing systems create an adequate environment for executing programs in parallel and getting the advantages of the computer resources distributed over the network. Message Passing Interface MPI is a portable model to build distributed systems and create an efficient environment for different scientific applications. Stream Control Transmission protocol (SCTP) is a modern transport layer protocol with new and efficient features that are not supported by TCP. SCTP combines the best features of TCP and User Datagram Protocol (UDP). It is message oriented protocol like UDP and provides effective algorithms for connection management like congestion control and flow control [2] [3].

In this work, we used SCTP as a transport layer protocol. We focused on using Multi-streaming and Multi-homing features in MPI applications and test the feasibility of using the aforementioned features on the performance of network throughput. An application programming interface API for MPI applications had been designed and implemented for this purpose[1][3][7][15].

The structure of this paper will be as follows. After introduction SCTP protocol and its features multi-homing and multi-streaming have been described then we discuss the most related projects and works to this paper. Then we describe and explain our work and different testing scenarios and the testing environment. Then, we discuss the results and the difference between the expected and the real results. Finally, the concluding section summarizes the article.

II. Stream control transmission protocol

SCTP is a transport layer protocols and combines the best features of the other transport layer protocols (TCP and UDP). It is connection-oriented, general purpose, reliable, message-oriented protocol. SCTP supports Multi-homing and Multi-streaming which represent the most interested features in modern Internet networks. Figure 1 shows SCTP position in Internet layer protocols [3].

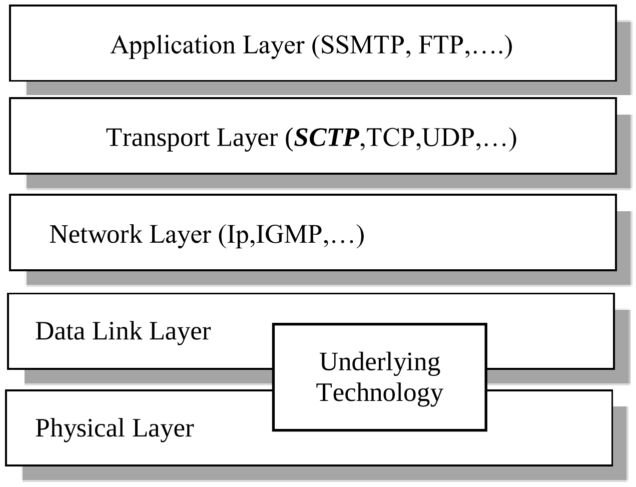
Application Layer (SSMTP, FTP,….)
Transport Layer (SCTP,TCP,UDP,…)
Network Layer (Ip,IGMP,…)
Data Link Layer
Underlying Technology
Physical Layer

Figure 1. TCP/IP Protocol.

A. *Multi-homing*

In computer networks, each endpoint uses one IP address for data exchanging even when it has more than one IP address and connected to more than one network. This case represents a weak point in performance of network communication especially with the revolutionary growth in hardware industry. Currently, each endpoint contains more than one network interface card, which means, it is possible to be connected to more than one network simultaneously. SCTP solves this problem by supporting Multi-homing feature. The communicating endpoints can exchange data via multi-paths using multiple IP addresses. Association (connection between SCTP endpoints) chooses one IP address and its path at each endpoint to create primary path for data communication while the other addresses and paths used as a backup. If the primary path suffered from any obstacle, the other paths will be used for data transmission. Current SCTP does not support concurrent data transmission over all available paths [1] [3].

B. *Multi-streaming*

The connection between SCTP endpoints is called association. SCTP supports Multi-streaming features in its endpoints association. Multi-streaming is one of the most powerful and important features of SCTP. TCP is a single stream protocol and the drawback in single stream protocols is that any blocking to this single stream hinders the transmission of data between communicating endpoints. This problem had been exceeded in SCTP Multi-streaming feature by creating multi streams for data transmission between the endpoints. Each stream is independent from other streams. If one of the streams faces any problems, data transmission continues in the other streams without any side effects.. For example, if three streams existed between two SCTP endpoints transmitting data and a data loss occurred in stream number three, only this stream needs to wait for the retransmission operation while the transmission over stream one and stream two will not be affected because SCTP does not need to warn about consecutive data delivery over all streams; just stream three should wait for retransmission of its lost packets [1][3][4].

III. RELATED WORK

Dreibholz [1] has investigated the property of parallel data transfer by using SCTP's extension CMT. Tests and analysis showed there are no big differences in the performance when using standard SCTP and SCTP-CMT. Also, the author described three drawbacks. Firstly, unordered delivery of data at receiver side. Secondly, the unordered data received by application layer at the receiver side and this cause a problem for the applications that needs ordered data. The third problem resulted from the unsuitable resources allocation for different application’s messages requirements. Kamal et al. [9] from the earliest projects suggested to use SCTP for MPI and used special module, request progressing interface RPI, to provide MPI's processes communication. Three methods were suggested and implemented to develop this RPI module to use SCTP instead of TCP. Iyengar et al. [10] Suggested utilization Multi-homing feature to implement concurrent multi path transfer of data. Three problems had been discovered as drawbacks associated to simultaneous transfer of data over multi paths, fast retransmission of already transmitted chunks at the sender side, reduced congestion window growth at the sender side and increasing of data traffic in the return path and solution to each one had been suggested. Penoff et al. [11] studied the feasibility of using Multi-homing and Multi-streaming features in MPI applications in computers clusters. They studied the effects of multi networks' configuration, messages scheduling, fault tolerance and congestion control on the cluster communication. Becke et al. [12] in their work developed and improved two load sharing transport layer protocols, SCTP-CMT and multipath TCP (MPTCP), and they compared the performance of those protocols by testing them over two networks, local network lab and real network between Germany and China over the internet. They discovered two important features that affects the performance of multi path networks, congestion control and path and resources management. Also, they suggested solutions depending on resource pooling to solve the problem of an inadequate use of the network's resources and proposed techniques to solve congestion control problem by splitting the congestion window to group of windows each one associated to one of the communicating paths.

IV. CMT FOR MPI: DESIGN AND IMPLEMENTATION

Recently, computational resources especially central processing units (CPUs) which are the main modules in the computational clusters became very high speed and performance efficient. Most of the current central processing units CPUs based on multi-core processors architectures with multi-level cache memory. In addition, some nodes in the computational clusters consists accelerators to support computational operations executed by CPUs. Computational accelerators include different devices and architectures like graphical processing units GPUs, field programmable gate arrays FPGAs. All these devices make cluster nodes massive computational devices, but on the other hand make the communication between these nodes the bottle neck of the total cluster performance. Efficient execution devices with unsuitable network communication can dramatically hinder computations achievement because this disparity in the performance. One of the good solutions to increase network communications is the use of multi-streaming communication with multi-homing feature. This solution utilizes all network interface cards available in cluster nodes to implement parallel data communication. This solution is the ideal one especially, for the currently established and in use clusters, because the replacement of the used network's infrastructure, topology and devices with a new technologies and infrastructures like infiniband is a hard and expensive solution. Also, increasing network throughput can decrease energy consumed by the clusters and this is a great benefit in this time. In this work, we designed an application programming interface API to utilize Multi-streaming and Multi-homing for concurrent multipath data transfer implementation. For this reason, multiple streams established between SCTP endpoints using independent networks and IP addresses for simultaneous data exchanging. These paths are disjoint and related to independent network.

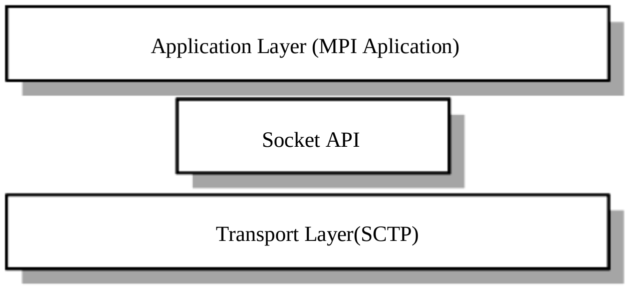

Figure 2. CMT Implementation in the Internet network Layers.

Standard SCTP establishes multiple paths between the communicated endpoints and uses one of these paths for data transmission and others as backup path. We used concurrent data transfer over multipath at the socket API layer, connecting socket between application and transport layers, not at the transport layer like aforementioned projects as shown in Figure 2. As mentioned before, standard SCTP uses one path as a primary one for data exchanging and the rest for backup but this API extends this feature to utilize all the established paths in data transmission.

The designed and developed API establishes independent transmission streams over different networks via network interface cards. Each message comes from the application layer, which is the upper layer, will be received by this API and then will be allocated to one of the established communication paths by implementing a Round-Robin scheduling algorithm to distribute the messages on the communication paths. This API receives user messages from application layer and distributes these messages over the communicating streams and start concurrent data transmission over the established paths. In this design each message that is received from application layer is allocated to specific resources and sent over independent network address interfaced to different networks.

Socket API is the middleware between the transport and application layers and able to utilize the resources of both of them and realizes the best usage of these resources. Working at this layer offers information on each type of messages created by different applications to allocate the required resources in dependence on the application's messages requirements making a better data exchanging performance. The designed API has been tested with four endpoints cluster with different network topologies. In the first test each endpoint interfaced to one network (single homed) while in the second test each one endpoint interfaced to two networks (dual homed). In the last test, endpoints interfaced to three networks (triple homed).

The testing environment had been built using OMNeT++ simulation environment and its framework Inet as shown in Figures 3-5. Each endpoint had been represented by an instance of the StandardHost module which is a module in OMNeT++'s INET framework. These endpoints used SCTP as a transport layer protocol. Each two endpoints had been connected by an instance of the Router module. The connection channels between the elements of the network are instances of the module DatarateChannel. In this module the data rate and the delay time of each channel could be changed to test the simulated network in different scenarios. The network was configured by using an instance of the module IPv4NetworkConfigurator to provide each endpoint with the appropriate IP address. MPI-NeTSim module was used to interface MPI applications to the simulated networks that was built by OMNeT++.

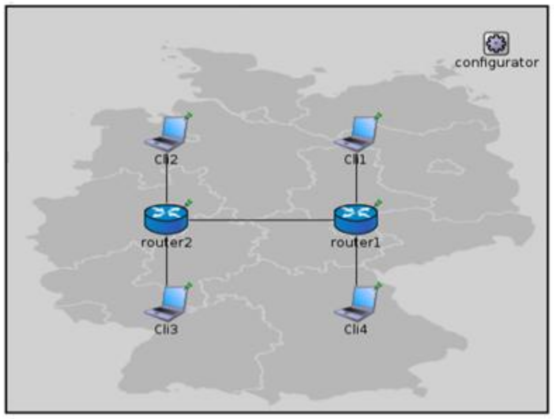

Figure 3. Single homed network.

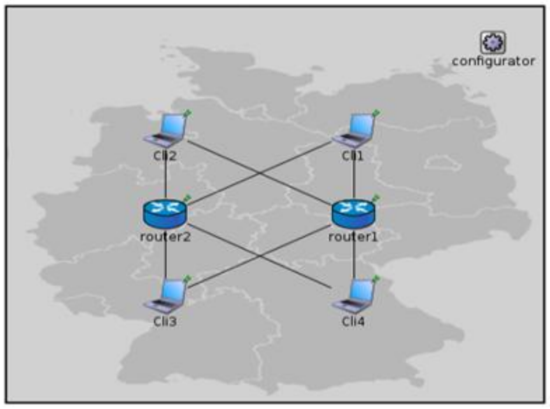

Figure 4. Dual homed network.

Figures 5-6 and Tables 1-2 show the performance of dual and triple homed cluster versus single homed cluster. We can clearly notice that the multi-homed clusters realize a better performance than the single homed and achieve higher data exchange rate leading to a faster execution time and better utilization of network resources.

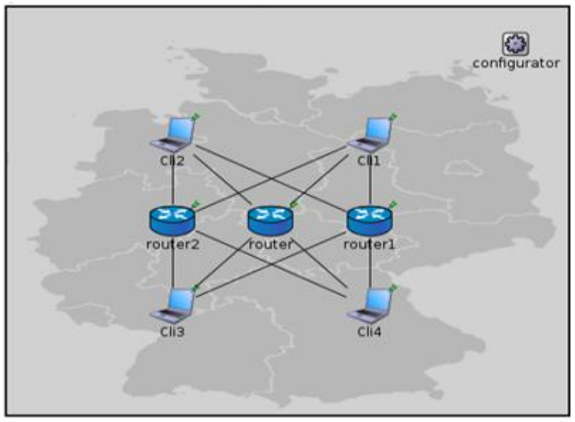

Figure 5. Triple homed network.

TABLE I. SINGLE AND DUAL HOMED NETWORK PERFORMANCE

Data Size MB	*Execution Time mSec*	*Execution Time mSec*
20	2.286814	1.351112
24	4.464083	3.578660
28	8.456284	6.659053
32	13.645028	10.097254
36	19.722385	14.629627
40	30.331633	19.855265

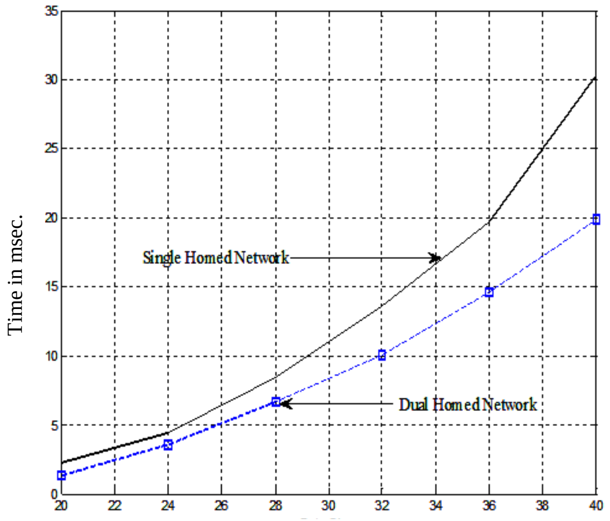

Figure 6. Singel versus dual homed network performance.

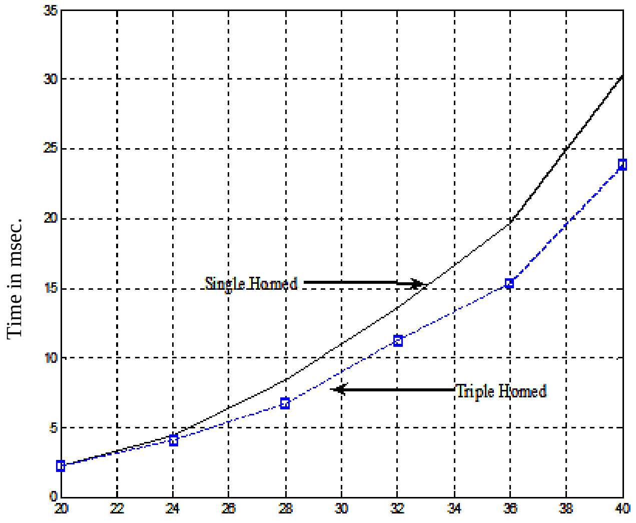

Figure 7. Singel versus Triple homed network performance.

TABLE II. SINGLE AND TRIPLE HOMED NETWORK PERFORMANCE

Data Size MB	*Execution Time mSec*	*Execution Time mSec*
20	2.286814	2.236239
24	4.464083	4.111785
28	8.456284	6.756544
32	13.645028	11.328705
36	19.722385	15.363163
40	30.331633	23.849022

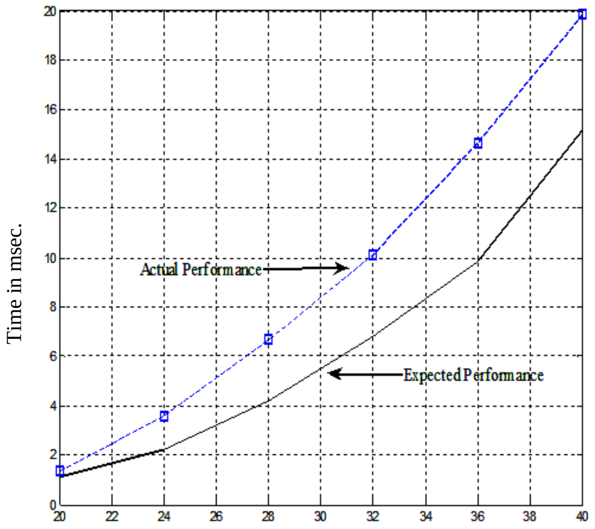

Figure 8. Real and expected performance of dual homed network.

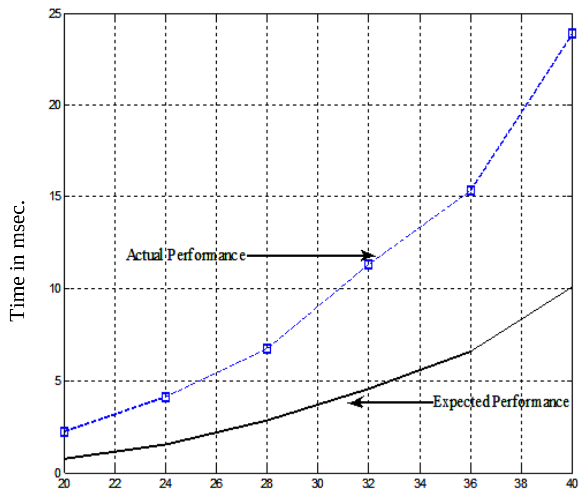

Figure 9. Real and expected performance of triple homed network.

At the same time, figures 8-9 show that we did not acquire the expected improvement in the performance because of increasing number of paths between endpoints should increase network throughput and decrease the time required for data exchanging but concurrent multipath transfer of data is associated to unnecessary fast retransmission of delayed data packets or few updates of sender congestion window which decrease the growth of this window and network throughput in addition to inadequate resources allocation for different messages requirements. These problems become more severe when increasing number of networks interfaced to cluster nodes and hinder the performance and cause these drawbacks in the designed API performance.

V. CONCLUSION

In this work, an application programming interface API has been developed to investigate the use of concurrent multipath data transfer in multi-homed computer clusters. A faster data transfer for MPI applications has been realized but stills lower than ideal performance that expected. The results showed also that the API performance decreases when the number of networks interfaced to endpoints increased.

The principal of concurrent multi path data transfer is expected to play a main role in different scientific computing applications to improve its performance. Also concurrent multipath transfer of data can play a main and essential role in high performance cloud computing like Hoopoe which is a GPGPU-based cloud computing, which is classified as a faster than real time infrastructure and has been used in different scientific applications [16] and needs a fast communication environment to prepare and transmits data with high data rates to integrates the work of this cloud computing. For this reason, the concurrent multipath data transfer will be the best solution to provide a required communication environment.

The most important point has been discovered in this work is that the increasing number of paths for data communication is not always the right solution to improve the performance and throughput of the network because concurrent data transfer is associated with problems of retransmission lost or delayed data packets and large congestion windows. As a result, to improve concurrent data transfer over multipath and different networks definitely it is needed to improve the total mechanisms of data packets management, retransmission policy and congestion window growth in SCTP protocol.

REFERENCES

[1] T. Dreibholz, "Evaluation and Optimisation of Multi-Path Transport using the Stream Control Transmission Protocol", 2012.
[2] H. Kamal, "SCTP-Based Middleware for MPI", 2005.
[3] B. A. Forouzan , "TCP/IP Protocol Suite", 2010.
[4] http://www.ibm.com/developerworks/library/l-sctp/
[5] F. Perotto, C. Casetti and G. Galante, "SCTP-based Transport Protocols for Concurrent Multipath Transfer", WCNC 2007, pp 2971-2976.
[6] A. Varga, OMNeT++ User Manual Version 4.3, 2011.
[7] M. Snir, S. Otto, S. Huss-Lederman, D. Walker and J. Dongarra, "MPI the complete reference", 1996.
[8] http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0 10174382002000100007.
[9] H. Kamal, B. Penoff and A. Wagner, "SCTP-based Middleware for MPI in Wide Area Networks",(CNCR05) 0-7695-2333-1/05 ,2005.
[10] J. R. Iyengar, K. C. Shah and P. D. Amer, "Concurrent Multipath Transfer Using SCTP Multihoming".
[11] B. Penoff, M. Tsai1, J. Iyengar and A. Wagner, "Using CMT in SCTP-Based MPI to Exploit Multiple Interfaces in Cluster Nodes", EuroPVM/MPI 2007, LNCS 4757, pp. 204–212, 2007.
[12] M. Becke, H. Adhari, E. P. Rathgeb, F. Fa, X. Yang and X. Zhou, "Comparison of Multipath TCP and CMT-SCTPbased on Intercontinental Measurements".
[13] http://artemis.wszib.edu.pl/~mradecki/prir/lab2/.
[14] B. Penoff, A. Wagner, M. Tuxen and I. Rungeler, "MPI-NeTSim: A network simulation module for MPI".
[15] J. R. Iyengar, P.l D. Amer and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths", IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 14, NO. 5, OCTOBER, pp 951-964, 2006.
[16] http://www.cass-hpc.com/solutions/hoopoe.